Team Orange's Cross-Topix

Bridging Music Silos using Semantic Technologies

X-Informatics Final Project, Spring 2011
Professor Peter Fox
Tetherless World Constellation
Rensselaer Polytechnic Institute

Team Members

Colin Anderson - Second semester MS in IT, MIS concentration
Yu Chen - 1st Year PhD ECSE, Tetherless World Constellation 518-522-7669
Samuel Johnson - 4th semester (part time) MS IT, HCI Concentration
Tim Lebo - 2nd Year Ph.D. Cognitive Science, Tetherless World Constellation
Amanda Olyha - Senior, Physics and CS

Contents

Appendix A: Project Pre-Definition

This is our group project for http://tw.rpi.edu/web/Courses/Xinformatics/2011

Slide 6 of http://tw.rpi.edu/media/latest/Xinformatics2011_week6.ppt

A) Analysis of existing information system content and architecture, critique, redesign and prototype redeployment

B) Pursuit of a detailed use case around a particular area of informatics, includes developing a prototype IS, architecture, design, etc.
•Due May 3 (write up) and May 10 (presentation)
•That's 7 (8) weeks
•Check in on progress in 3 weeks

Did Peter outline what we would need to "evaluate", or would we come up with that criteria?

Evaluate system ideas

PROPOSAL E.1

But perhaps "evaluating an existing system" would be easier b/c it is concrete and we can poke it.

Perhaps we could evaluate the VIVO system. http://vivoweb.org/
It is a professional research "social networking" infrastructure that uses RDF as its representation.

Use case ideas

PROPOSAL U.1

Tim ran into some poorly managed MRI handling that we could develop a use case for.

Meetings

Interest in music.
Amanda is the orchestra librarian.
Safari books - hard to navigate. Went flash and it is terrible.

Resources

Sheet music - physical stuff. Copyright.
Libraries
http://www3.cpdl.org/wiki/
http://imslp.org/wiki/

LOD cloud of music. Music Genome Project (Pandora)

http://richard.cyganiak.de/2007/10/lod/lod-datasets_2010-09-22_colored.html

<http://tw.rpi.edu/instances/TimLebo> a foaf:Person .

http://tw.rpi.edu/web/person/TimLebo

http://validator.linkeddata.org/vapour

http://en.wikipedia.org/wiki/Composer

http://dbpedia.org/resource/Composer ←----

validating dbpedia's URI for composer:

http://validator.linkeddata.org/vapour?vocabUri=http%3A%2F%2Fdbpedia.org%2Fresource%2FComposer&classUri=http%3A%2F%2F&propertyUri=http%3A%2F%2F&instanceUri=http%3A%2F%2F&defaultResponse=dontmind&userAgent=vapour.sourceforge.net

http://www3.cpdl.org/wiki/index.php/Emanuele_d%27Astorga

2 questions to ask:
1) fitting requirements of project
2) can we do it in a month.

Appendix C: Proposals

done: everyone write 3 paragraphs proposing what we do. (Due Sunday evening) - for review by team on Monday.

- What resources we will use
- What benefits we will provide.
- How we conform to each request in the assignment.
- What technologies we will use
- The kind of user would benefit
- How what we'd do fits into the topics discussed in class
- what deliverable we will provide

Proposal by Yu:
We are trying to redesign the digital music library, after analyzing a set of online music libraries, leveraging the social, cognitive and domain concerns that could make a music information system much easier accessible for the music fans. The music libraries that we reference are those such as IMSP-Pertrucci Music Library, Free Choral Sheet Music etc.

The reason why we redesign the online music library is based on the observation that the pages are information-overloaded and the navigation is neither intuitive nor effective. These are something we are trying to modify and optimize. In a word, we will redesign the music information system by providing more user-friendly interface that better navigate the user to the piece of information that required. We will also make better arrangement of the presentation of the pages such that the user could immediately get what the pages tells them in a short time.

As we see the requirements in the assignment, we would definitely draw the diagram of the conceptual model again that better illustrate the infrastructure of all the information system is found in components. What's more, we will implement a simple prototype of the system within web browser to illustrate our solutions.

Technologies that we might use are Flash and Action script, PHP etc. All that related to UI design might help. (I independently designed and implemented several Flash and Action scripts apps before, hope it could help in designing our new UI)

The users that could benefit from our design could be more than the professional musicians. Music lovers without so much domain expertise could also find the scores or recordings.

To fit into the topics of the materials covered in class, we need definitely consider cognitive, semiotics and social concerns towards a good information system. Therefore, we each might need to be responsible for a particular section of the concerns mentioned in class. In

implementing the system, we should collect the suggestions from each of us and realize the functionalities accordingly.

A deliverables that I could think of is a demo Flash application or a set of linked webpages, which shows how user could better get the information according to our re-newed schemes.

Proposal by Sam:
I actually found what Yu wrote to be in line with what my thoughts were, so I'm going to attempt to extend and flesh-out what he wrote, rather than attempt to come up with something completely new (my changes/additions in *italic*):

Proposal by Yu (*with extensions/additions by Sam*):
We are trying to redesign the digital music library, after analyzing a set of online music libraries, leveraging the social, cognitive and domain concerns that could make a music information system much easier accessible for the music fans. The music libraries that we reference are those such as IMSP-Pertrucci Music Library, Free Choral Sheet Music etc.
I'd propose focusing on professional or amateur musicians, rather than music fans. The typical music fan isn't interested in the sheet music itself so much as the product of that sheet music. Because the libraries we discussed on Friday are libraries of sheet music rather than recordings of musical performances, I believe that we should assume any users of our redesigned system would be musicians intending to use the sheet music, rather than fans looking for music to "consume."

The reason why we redesign the online music library is based on the observation that the pages are information-overloaded and the navigation is neither intuitive nor effective. These are something we are trying to modify and optimize. In a word, we will redesign the music information system by providing more user-friendly interface that better navigate the user to the piece of information that required. We will also make better arrangement of the presentation of the pages such that the user could immediately get what the pages tells them in a short time.
I'm less concerned with the interface itself than the way the information is tagged and organized. We'll need to come up with a few use cases specifying the reasons a musician comes to one of these sites looking for sheet music, and focus on helping that user achieve his or her goal.

As a semi-professional musician myself, I can both add to our domain knowledge in creating these use cases, as well as provide contacts to professionals in the area who rely on the databases as they currently exist. That will help us when we create the use-cases the project requires.

As we see the requirements in the assignment, we would definitely draw the diagram of the conceptual model again that better illustrate the infrastructure of all the information system is found in components. What's more, we will implement a simple prototype of the system within web browser to illustrate our solutions.
Again, my (and Amanda's) experience within the domain will be useful in creating a set of

diagrams modeling the information. This will be particularly interesting when contrasted with people who have little experience with music--the combination could potentially be very effective.

There are any number of different potential attributes for any given music, and I think that will give a rich complexity to the problem that will make it challenging. While that's what makes it worthwhile, we should also be aware that it won't be straightforward. I'm interested to talk more about what Tim started talking about, because I'm not totally sure I understood it. For any given score, there's an immense amount of available meta-data that could be associated with it, and finding ways to make that meta-data accessible by the users, whether through better searching, or better organization, would be a good direction to go.

Technologies that we might use are Flash and Action script, PHP etc. All that related to UI design might help. (I independently designed and implemented several Flash and Action scripts apps before, hope it could help in designing our new UI)
I don't have any knowledge of Flash or ActionScript (I'm too cheap to pay for Adobe's development tools); my UI language of choice is JavaScript, but I'm sure we can find a way to sort out that difference.

The users that could benefit from our design could be more than the professional musicians. Music lovers without so much domain expertise could also find the scores or recordings.
As I said above, I don't know that I agree with this, but I'd be interested in hearing more of how a non-musician would make use of musical scores. We need to be clear about whether we're including recordings within our domain. I believe the only actual sound files that are available on these sites are MIDI versions of the musical scores.

To fit into the topics of the materials covered in class, we need definitely consider cognitive, semiotics and social concerns towards a good information system. Therefore, we each might need to be responsible for a particular section of the concerns mentioned in class. In implementing the system, we should collect the suggestions from each of us and realize the functionalities accordingly.

A deliverables that I could think of is a demo Flash application or a set of linked webpages, which shows how user could better get the information according to our re-newed schemes.
We might also want to consider what Tim talked about i.e., finding ways to integrate these databases into the existing semantic web, which would enable much broader use by the academic community, as well as the application of existing informatics tools and techniques in the searching/browsing of the musical scores. I'd be interested to hear from Tim more about how that could work.

Proposal by Amanda:

I agree with most of what has been said above, but there are a few things I would like to add:

Like Sam had said above, I think the users of this redesigned system will probably be for professional or amateur musicians, although we may not want to only limit ourselves to this. We can determine the specifics once we create our specific use cases.

With this in mind, we should have a strong, well-defined search use case planned out since it will be one of the main points of entry to the system. Along with a search, I think the browse options should also be well-designed. Because of all the metadata that we could potentially use, browsing can be a very strong feature.

Colin's addition:

I agree with much that has been said above so I will add my two cents for what its worth.

I agree with Sam and Amanda that the functionality of the system should be our main concern. However, I would not disregard the UI as extraneous. Yu's expertise here could make the system more popular and easier to use for all involved. Before we accept or disregard technologies to use as well, we need to get together and decide what is best for the project.

I think we could look at this as an educational tool as well. We are focusing on amateur and professional musicians, but we could also include music instructors (choral, band, orchestra) to the mix as well.

I am concerned with the "Use Case" itself. Reading the instructions for the final assignment, the use case needs to be be well defined and the scope of this project needs to very tight. Based on what I have read, and what I thought the use case should be, we really need to nail that down at Monday's meeting.

Proposal by Tim:

- Steps
    - Install a local Mediawiki
        - Install the Semantic Mediawiki (SMW) extension
        - copy several sample values to local wiki
        - annotate using SMW markup
        - associate annotated pages to LOD cloud and ontologies
        - demonstrate local queries
        - import appropriate ontologies (including FRBR)
        - access the data as RDF dumps.
    - Contact existing wiki owners and convince them to install the sem media wiki extension
        - demonstrate markup
        - train community
        - demonstrate queries

        - ask them what problems they have and document them.
    - explore LOD music
        - establish linking among existing cloud data via the sem media wiki
        - accumulate some of the LOD into a sparql endpoint
        - accumualte some of the wiki RDf into same sparql endpoint
    - crawl existing mediawikis and try to grab some structure
        - encode as RDF establishing URIs as if they had a SMW installation.
    - index physical sheet music
        - don't publish, just provide metadata and pointers to where it is phsyciall and whom to contact.
    - Additional UI
        - javascript?
        - actioncript?
- Resources:
    - a server machine with admin access to host web stuff
    - SMW software
    - RDF tools/crawlers
    - Music ML?
- Benefits:
    - answering new questions
    - person finding music would benefit
    - linking to Library of Congress?
    - I like Sam's idea of focusing on musicians proper.
- Deliverables
    - use case documentation
    - prototype system
    - people saying we are awesome and helped them.
    - music ontology

Appendix B: Meeting Notes

Meeting notes: Tuesday 2011 April 05

Instructor: time period, "no half notes"

independent musician: I want sheet music. arrangements.

create an ontology? keep it small

emotional component - music at fourth of july, at a wedding.

Meeting notes: Thursday 2011 April 07

Tonight's Goals
1. Subset of Wiki
2. Develop example content
3. How search that?
4. Small model

    1. We want to Facilitate Browsing
    2. Be able to address number of instruments

Browsing, NOT search.
Don't know composer, style, piece.
Upbeat music for marching band.
Given a set of instruments, what works?

Meeting Notes, Thursday 2011 April 07

http://imslp.org/wiki/Aire_(with_variations)_(Babell,_William)
http://www3.cpdl.org/wiki/index.php/A_Third_Set_of_Psalm_Tunes_%28Thomas_Clark%29

Meeting Notes, Thursday 2011 April 21

Discussed progress of work

Yu – created all diagrams for the project, created SPARQL endpoint
Amanda – Created Use cases, ran two cases for similarities
Colin - developed use cases
Tim – Organized the data set management design, reviewed and modified document, triple store installation with Sir Patrick West, coordinated Sparkle with Sir Patrick West and future Doctor Chen (Tim is our master)
Samuel – Finished implementing the joiner, and developed a read me (very lengthy assignment)

Tonight's work (done during meeting)

Yu – Testing triples on SPARQL
Amanda – Coming up with histogram of similarity measure
Colin – Taking notes, documenting work, beginning paper
Tim –
Samuel -

Next steps (after meeting)
Yu

- vote RDF example - put onto google doc
- ARC2 post and query to verify - put on google doc

- secondary: activity diagram for use case

Amanda

- histogram of 10*10 DONE
- run on full 1B pages
- future work
- propose threshold and resulting subset size
- starting PHP See Also widget that accepts URL of a wiki page, queries the endpoint for suggestions, and displays it

Samuel

- start social machine UI
- execute SPARQL query

Tim

- design SPARQL query that social machine UI will execute
- design SPARQL query that See Also page query
- verify ARC2 named graph population
- verify Samuel's similarity output

Colin

-

Semantic MediaWiki: extra syntax give a little semantics:

```
===Publication===
[[composer::Thomas Clark]]'s ''A Third Set of Psalm Tunes with a
Magnificat Nunc Dimittis and an Anthem; with an Instrumental Bass''
was published in London by Button & Whitaker of 75, St Paul's Church
Yard. The collection is undated but the ''Hymn Tune Index'' notes
that the dates of activity of the publisher at this address were
c1808–14, and suggests that this book dates from c1809.

{{#ask:
  [[composer:Thomas Clark]]
|?title
|?date
}}
```

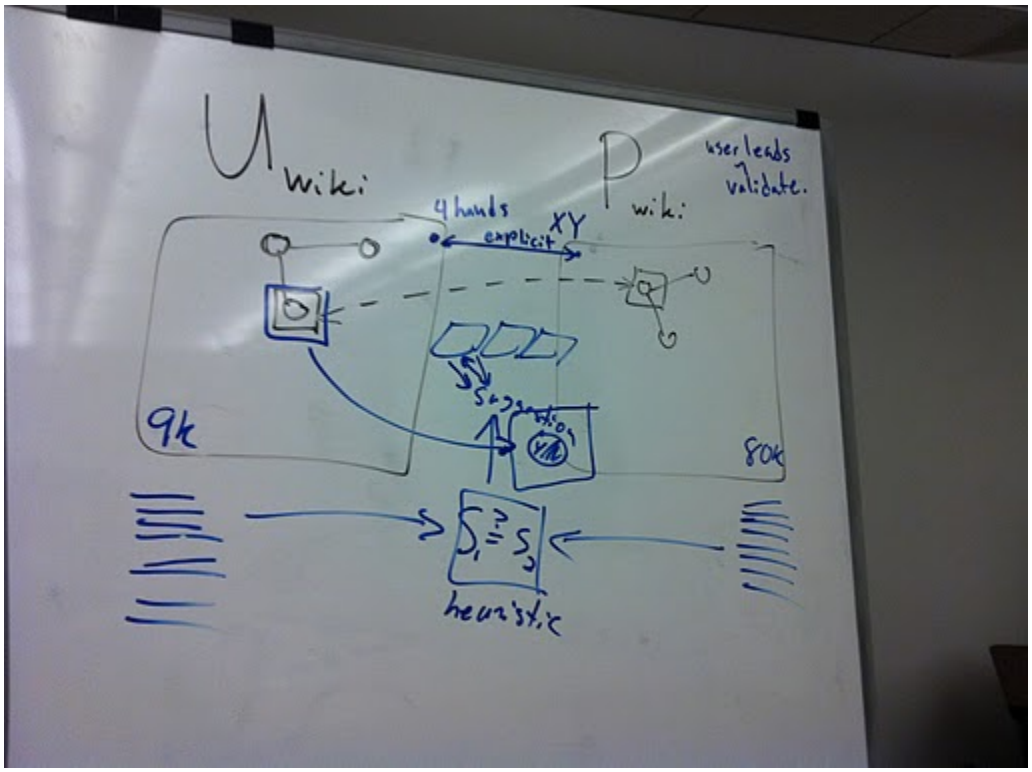
The following link to the same piece:
Pretty: http://imslp.org/wiki/Christ_lag_in_Todesbanden,_BWV_4_(Bach,_Johann_Sebastian)
Ugly: http://www3.cpdl.org/wiki/index.php/Cantata_BWV_4_-_Christ_lag_in_Todesbanden_(Johann_Sebastian_Bach)

List of potential attributes that we'd want to search on any particular musical score.

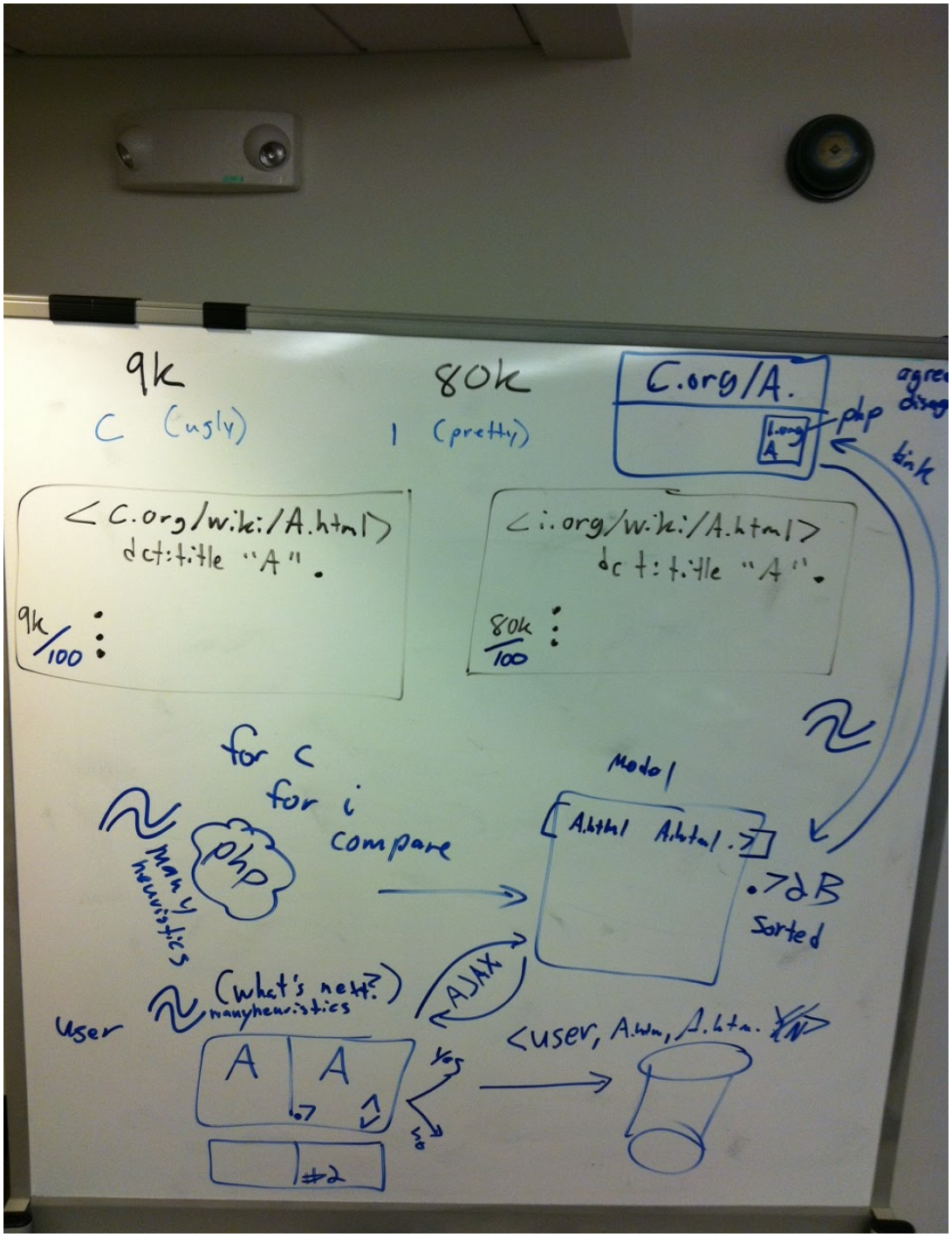
http://data-gov.tw.rpi.edu/wiki/Tim_Lebo

http://www3.cpdl.org/wiki/index.php/Daniel_Vetter cites a Wikipedia article ADD TO FUTURE WORK (Linking via wikipedia)

Project Definition and Task Decomposition

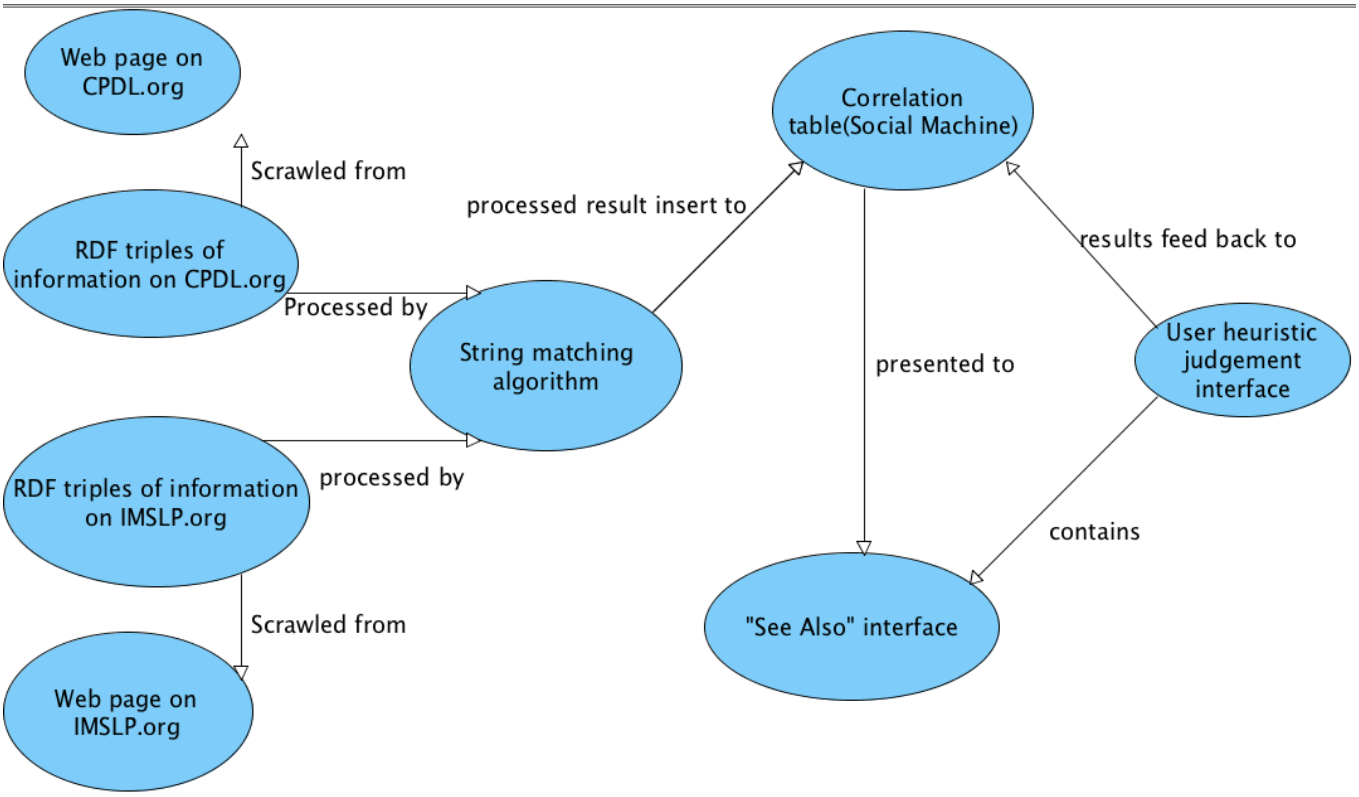


Concept Model v1.2011.04.07



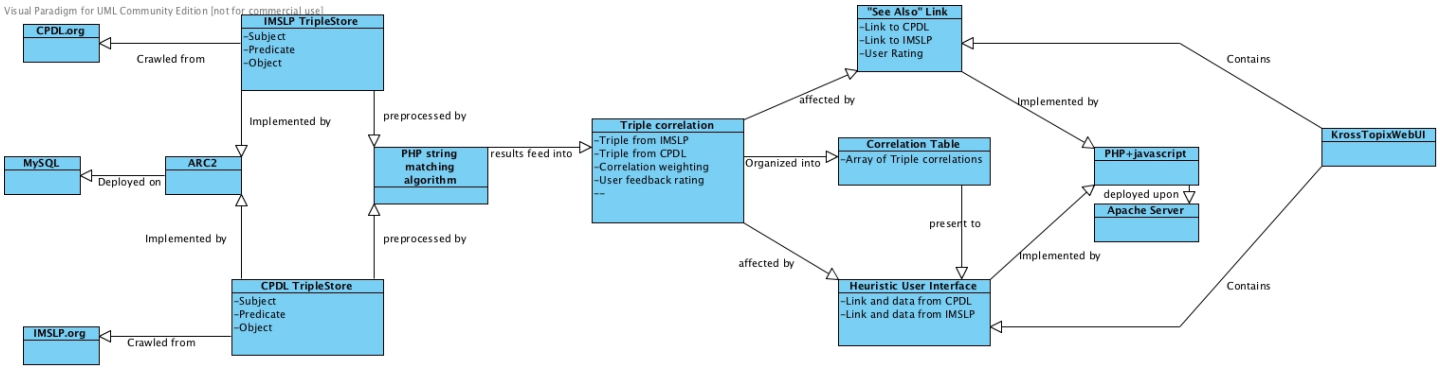
Concept Model v2.2011.04.14

Concept Model:



Concept Model v3.2011.04.14

Logical Model:



Logical Model v1.2011.04.14

Everyone get a github account

- (X) Tim
- (X) Sam
- (X) Yu
- () Colin TODO
- (X) Amanda

Set up version control

(X) Start repository (Sam)
Git-hub link:
git://github.com/samuelbjohnson/cross-topix.git
git@github.com:samuelbjohnson/cross-topix.git

```
bash-3.2$ mkdir git
bash-3.2$ cd git
bash-3.2$ git clone git@github.com:samuelbjohnson/cross-topix.git
bash-3.2$ cd into cross-topix/
<edit helloWorld.html - add your name>
bash-3.2$ git remote add [repo name] git@github.com:samuelbjohnson/
cross-topix.git
bash-3.2$ git status
bash-3.2$ git diff helloWorld.html
bash-3.2$ git add helloWorld.html
bash-3.2$ git commit helloWorld.html -m "Some message"
bash-3.2$ git push [repo name] master
```

Scrape pages on two wikis

(X) Scrape URLs and titles. (Tim)

- Conceptual Design of Page Title Gathering (pdf on github)

Scrape Starting Point	Scrape Results	**Source** Identifier	**Dataset** Identifier	**Version** Identifier	Dataset URI
Special:AllPages	imslp.ttl	imslp-org	wiki-pages	2011-Apr-14	2011-Apr-14
Special:AllPages	cpdl.ttl	cpdl-org	wiki-pages	2011-Apr-14	2011-Apr-14

(X) Design Crawl encoding using RDF. (Tim)

- http://dublincore.org/documents/dcmi-terms/#terms-title

(X) Encode scrape results using the Turtle syntax. (Tim)

The following Turtle syntax for our RDF abstract model shows an example of the data provided by the scraper. The *subjects* of the triples are the pages themselves, the dcterms:title *predicate* is being reused, and the *object* of the triple is the title as listed on the wiki's index page.

```
@prefix dcterms: <http://purl.org/dc/terms/> .

<http://www3.cpdl.org/wiki/index.php/20th_century>
   dcterms:title "20th century" .

<http://www3.cpdl.org/wiki/index.php/21st_century>
   dcterms:title "21st century" .
```

NOTE: The use of *triple double quotes* (“””) in the Turtle syntax is legitimate, but ARC2's parser does not handle it properly, so we needed to backtrack and use only *single double quotes* (“).

Apply String Matching Algorithm

TODO: Apply string matching heuristics
http://en.wikipedia.org/wiki/Levenshtein_distance
() Survey string matching implementations (off the shelf)

- http://php.net/manual/en/function.levenshtein.php
- TODO: the java one list here

(X) Implement joiner (Samuel)

- comparisonModuleJava/Joiner
- (X) write README that covers usage (How do I run it?)
- comparisonModuleJava has a jar file
    - Copy the jar file to the directory where you want to run it
    - `java -jar joiner.jar [firstFileName [secondFileName]]`
    - output is in `data.ttl`

Java code to create MD5 of concatenation of SPO:

```
byte[] bytesOfMessage;
try {
              // http://stackoverflow.com/questions/415953/generate-md5-hash-in-java
              bytesOfMessage   = (myInstanceLocalPrefix + myLocalName).getBytes("UTF-
8");
              MessageDigest md = MessageDigest.getInstance("MD5");
              byte[]     digest   = md.digest(bytesOfMessage);
              BigInteger bigInt   = new BigInteger(1,digest);
              String     hashtext = bigInt.toString(16);
              System.out.println(hashtext + " -> " + hashtext.substring(0,3));
} catch (UnsupportedEncodingException e) {
              e.printStackTrace();
} catch (NoSuchAlgorithmException e) {
              e.printStackTrace();
}
```

Java code to abbreviate the long URIs with prefixes in the Turtle serialization: TODO

```
// conn is your RepositoryConnection from Repository.getConnection();
conn.setNamespace("comparison","http://beta.twc.rpi.edu/id/cross-topix/alpha/");
conn.setNamespace("xt",         "http://purl.org/twc/vocab/cross-topix#");
conn.setNamespace("xsd",        "http://www.w3.org/2001/XMLSchema#");
```

The namespace for this project's vocabulary was chosen to align with the purl.org service, which provides a redirection service when its URIs are requested.

- http://purl.org/twc/vocab/cross-topix# is the namespace for the ontology;
- dereferencing it resolves to the file in our github https://github.com/samuelbjohnson/cross-topix/raw/master/ontology/vocab.ttl

- WWW user interface for the file is available at https://github.com/samuelbjohnson/cross-topix/blob/master/ontology/vocab.ttl

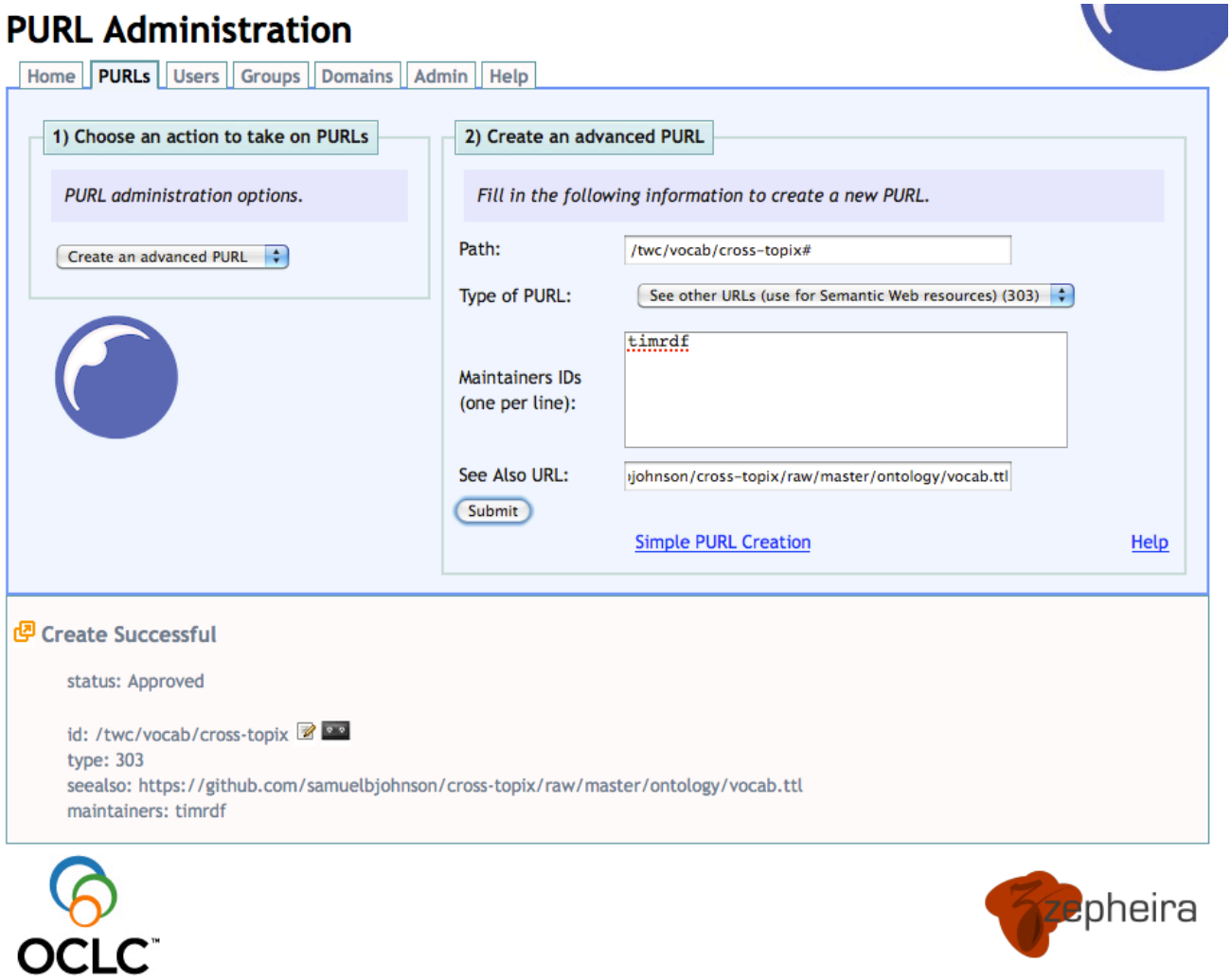


Figure: This screenshot documents the creation of the purl.org vocabulary namespace and its redirection to the version-controlled OWL document on our github repository. This enables anyone to dereference URIs in our vocabulary to obtain a formal description for how those terms should be interpretted. It also allows us to control the development of the vocabulary while effortlessly allowing others to access it according to semantic web principles.

() Design output encoding (cite algorithm, give rating for string pairs)

- title_u
- title_p
- similarity
- heuristic used

TODO: prepend “md5” before the md5 value, so we can abbreviate:

```
@prefix xsd:        <http://www.w3.org/2001/XMLSchema#> .
```

```
@prefix dcterms:    <http://purl.org/dc/terms/> .
@prefix xt:         <http://purl.org/twc/vocab/cross-topix#> .
@prefix comparison: <http://beta.twc.rpi.edu/id/cross-topix/alpha/> .

comparison:551c2c8a0c2a2e07b488d1b8110c116f

   xt:comparable_1 <http://imslp.org/wiki/Ave_verum_corpus,_K.618_(Mozart,_Wolfgang_Amadeus)> ;


xt:comparable_2 <http://www3.cpdl.org/wiki/index.php/
Ave_verum_corpus,_KV_618_(Wolfgang_Amadeus_Mozart)> ;

   xt:similarity "0.36"^^xsd:double
.
```

Establish Ground Truth

(X) Amanda hand-selected links between the two wikis, which provides 10 test cases.
() Sam hand select 10 test cases (links) TODO
The following table shows the datasets containing the hand-selected associations among the two wikis. The **Version** identifier was selected based on when it was completed.

Crawl	Source	Dataset	Version	Dataset URI
amanda.ttl	orange-amanda	ground-truth	2011-Apr-19	2011-Apr-19
TODO	orange-samuel	ground-truth	2011-	

(X) Run hand-picked test cases through String Joiner (Amanda)
The following table shows the results of the String Joiner applied to the hand-selected ground truth datasets listed in the previous table.

Crawl	Source	Dataset	Version	Dataset URI
results_amanda.ttl	orange-joiner	title-similarities	2011-Apr-20	2011-Apr-20

() Correct Turtle syntax output. - TIM
() Does the String Distance measure correlate to our desire of Wiki Page similarity?

(X) Create histogram of similarity measure for 10+10 dataset (Amanda)
() Create histogram of similarity measure for 1 B dataset (Amanda)

Histograms

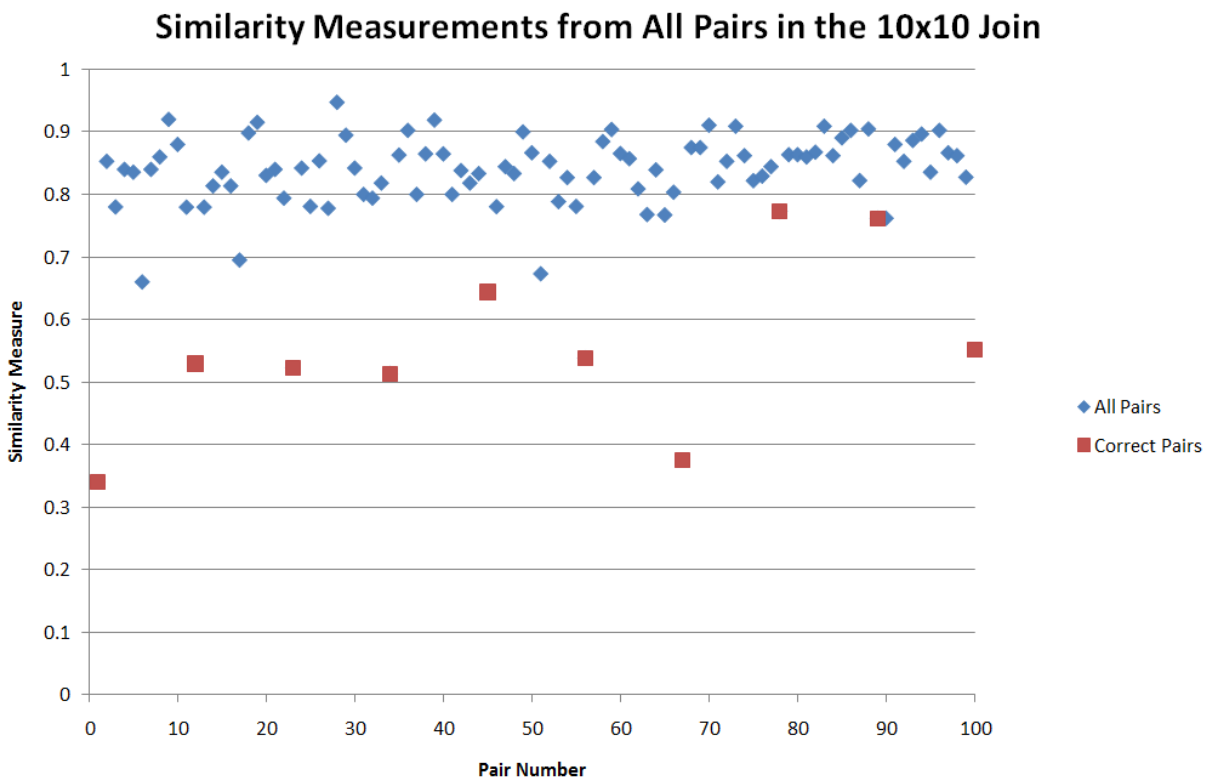


Figure: Values of the similarity measures for all pairs created by the joining of the 10-count sets. The x axis is irrelevant - it is the “pair number” (so the first pair to be compared is 1 the second pair is 2, etc.). The y axis is the actual value returned by the function. See the chart below for a histogram broken up in 0.1 increments.

These values were calculated by comparing 10 pages from IMPSLP.org with 10 pages from CPDL.org.

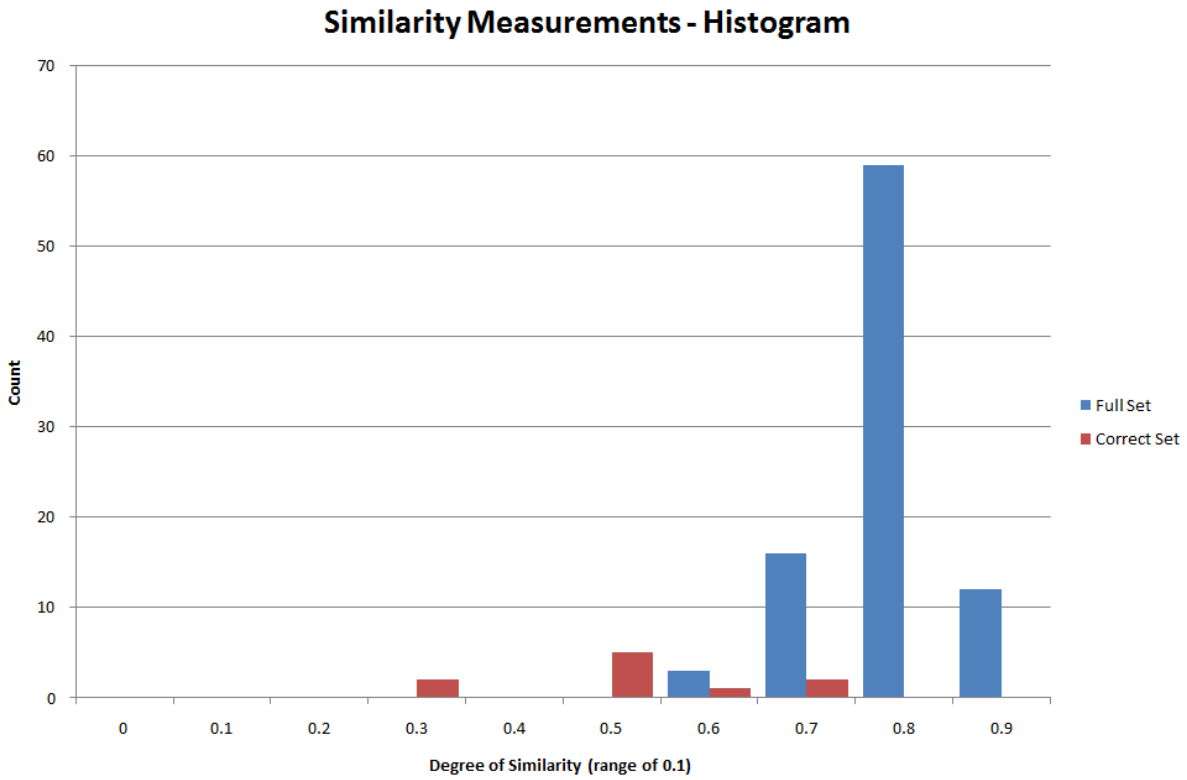


Figure: Histogram of the similarity measurements from the 10 x 10 hand-picked examples. The red bars are the correct matchings while the blue bars are the all the others as a result of the join. The values are broken into 0.1 value increments, with the displayed number being the lower bound.

Can you paste these into the google doc and write up a little about them (how they were created, how to read it, what they mean, and why we care)? We need to convince ourselves and our audience that this similarity measure will answer the mail.

String Matching Algorithm Improvements

To determine if two page titles are similar, we are currently calculating the Levenshtein Distance (also known as the Edit Distance) between the two. This gives us a number that represents the minimum number of characters that need to be replaced for the two strings to be considered equivalent. Once we normalized this value to take into the account the length of the strings, we obtained a value that gives us a simple metric to determine whether two titles are similar (and to what degree).

Although this approach works well for a good number of cases, it can be improved to obtain more accurate results. One way to do this would be to take advantage of the format that these two wikis store their information. IMSLP.org, for example, stores all their composers with their last name followed by their first name, while CPDL.org does it the other way around. By

calculating the “edit distance” of two composers, we will get a value that is much higher than we would want simply because we are comparing the words out of order. If there was some way to care the titles on a word-by-word basis rather than as a whole, we can create a more accurate similarity measure.

Another point of improvement is for pages that are for specific composers or genres. IMPSL.org prefixes all pages of this type with a “Category:” tag which CPDL.org does not do. Since these extra characters are considered in the calculation of the similarity, our values may be artificially high. If we ignored “Category:” tags, we might also be able to gather more accurate similarity values.

Install Triple Store and SPARQL Endpoint

(X) Install Triple Store and SPARQL Endpoint (Yu; Thanks Patrick!)

- https://github.com/semsol/arc2/wiki
- Human HTML interface is at http://leo.tw.rpi.edu:81/endpoint.php
- **http://leo.tw.rpi.edu:81/endpoint.php** is also the Web Service
- NOTE: the endpoint is ONLY accessible from RPI's network. Use VPN if off campus.
- http://www4.wiwiss.fu-berlin.de/bizer/rdfapi/tutorial/netapi.html was NOT used.
- Reference pages:
    - http://bnode.org/blog/2007/11/26/load-insert-and-delete-in-arc2-via-sparql-plus
    - https://help.ubuntu.com/community/ApacheMySQLPHP
    - https://github.com/semsol/arc2/wiki

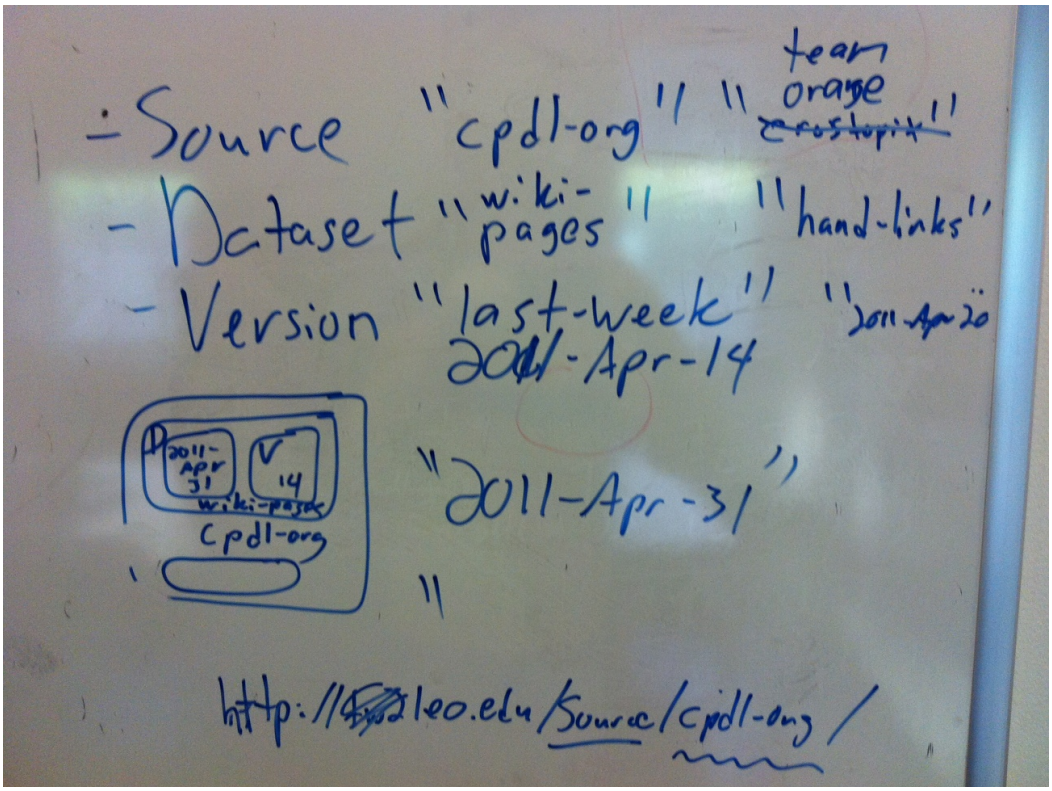


Figure: Illustration of the "three-attribute" contextualizing naming conversion developed by Lebo, Williams, and Graves in the LOGD project. Providing short identifiers for **source**, **dataset**, and **version** allows for the construction of a dataset's URI. Our current project has two external sources (**cpdl-org** and **imslp-org**) and several internal sources (**team-orange, orange-amanda, orange-joiner**).

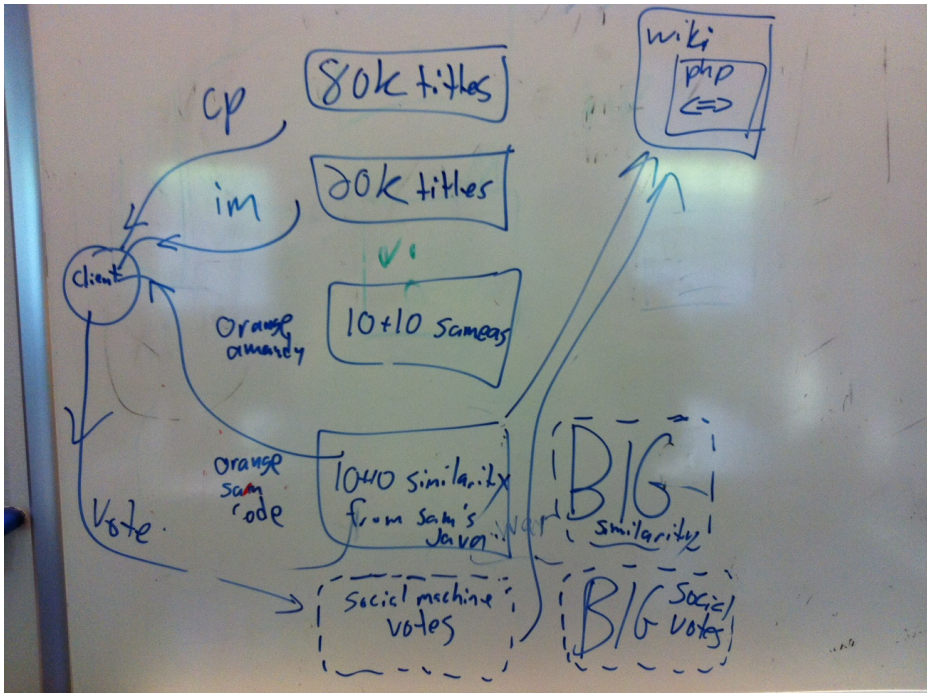


Figure: Whiteboard rendering of the first four named graphs populated in our yet-to-be-installed triple store. "***80k titles - cp***" is the page-titles obtained from scraping CPDL.org, "***20k titles - im***" is the page-titles obtained from scraping IMSLP.org. "***10+10 sameas - orange-amanda***" is a hand-curated list of pages that refer to the common concepts. "***10+10 similarity from Sam's Java - orange-samecode***" is the similarity measurements of Amanda's hand-curated list. The client queries across all three of these to request a confirmation or invalidation from the social machine human, whose vote is placed back into a separate named graph ("***social machine votes"***). The "***wiki php < = >***" produces an HTML widget that suggests links to the second wiki within the wiki pages of the first. The "***BIG similarity***" and "**BIG social votes**" are the larger analogues using the full 80k and 20k page-titles datasets, which we expect to be too large.

() load 80k titles and 20k titles into named graphs named after their dataset URI
() load ground truth files into "
()

Named graph design strategy:

- Data loaded from TEAM ORANGE will be loaded from our github repository. The named graph will NOT correspond to the URL from which it was retrieved. Instead, the named graph will be named according to the "Source, Dataset, Version" Dataset URI convention.
- Data loaded from external sources are placed into named graph corresponding to the URL of the original document.

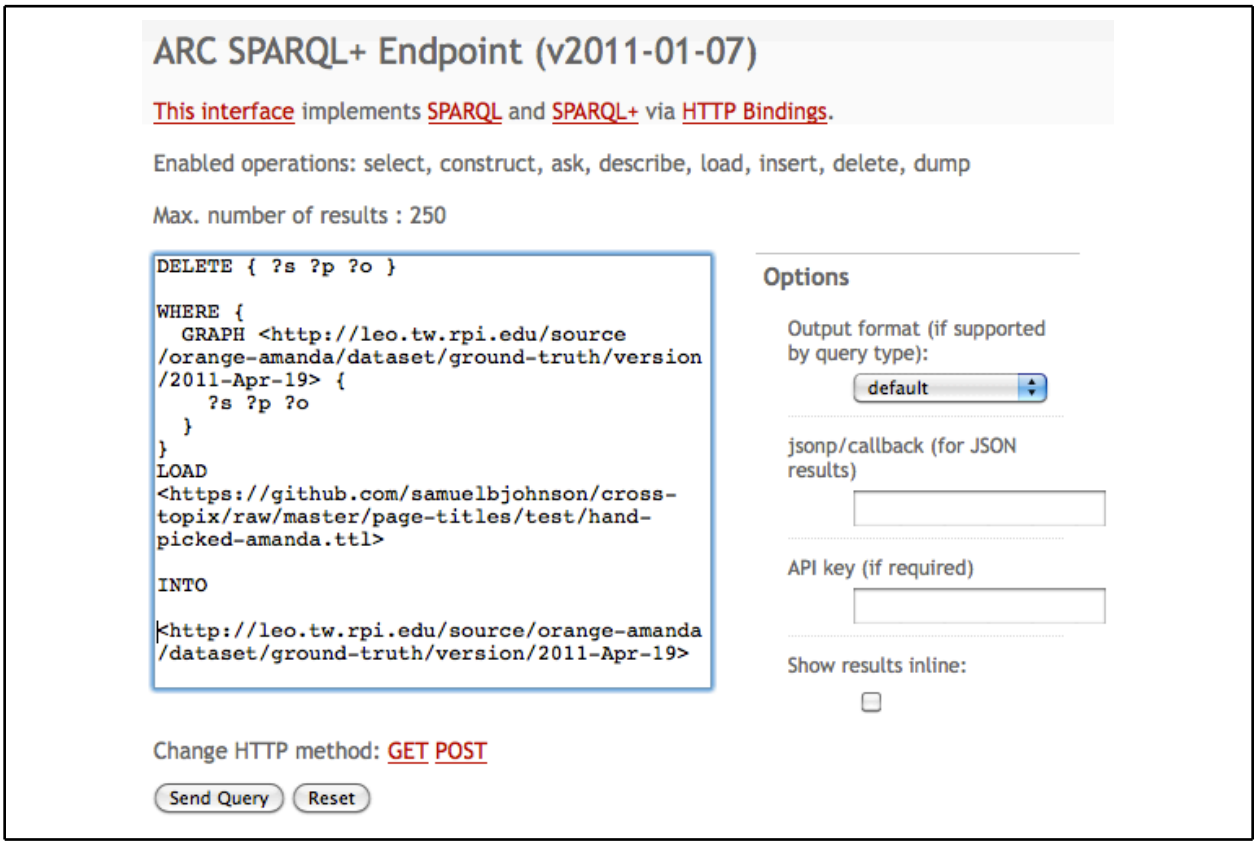
Load all of the RDF for the **ground truth use case** from the github repository into the endpoint:

```
# NOTE: These are NOT sample queries. These are the queries we use to populate the endpoint.

#
# The titles of Amanda's 10+10 sample use case
#
DELETE { ?s ?p ?o }
WHERE { GRAPH <http://leo.tw.rpi.edu/source/orange-amanda/dataset/ground-truth/version/2011-
Apr-19> {
?s ?p ?o }
}
LOAD
<https://github.com/samuelbjohnson/cross-topix/raw/master/page-titles/test/hand-picked-
amanda.ttl>
INTO
<http://leo.tw.rpi.edu/source/orange-amanda/dataset/ground-truth/version/2011-Apr-19>

#
# Similarities among Amanda's 10+10 sample use case
#
DELETE { ?s ?p ?o }
WHERE { GRAPH <http://leo.tw.rpi.edu/source/orange-joiner/dataset/title-similarities/version/
2011-Apr-20> {
?s ?p ?o }
}
LOAD <https://github.com/samuelbjohnson/cross-topix/raw/master/string-matching_tests/
results_amanda.ttl>
INTO <http://leo.tw.rpi.edu/source/orange-joiner/dataset/title-similarities/version/2011-Apr-
20>
```

Screenshot of ARC2's Web Browser HTML interface that accepts queries:



What named graphs are in the endpoint (results) (use GET)*?*

```
select distinct ?g
where {
  graph ?g { ?s ?p ?o }
}
```

Deleting triples in all named graphs (must be POST):

```
delete {
   ?s ?p ?o
}
where {
   graph ?g { ?s ?p ?o }
}
```

What are the titles of the wiki pages that Amanda hand-selected? (results)

```
prefix dcterms: <http://purl.org/dc/terms/>
prefix cpdl:   <http://www3.cpdl.org/wiki/index.php/>
prefix imslp:   <http://imslp.org/wiki/>

SELECT ?page ?title

WHERE {
  GRAPH <http://leo.tw.rpi.edu/source/orange-amanda/dataset/ground-truth/version/2011-Apr-19> {
       ?page dcterms:title ?title
  }
}
```

What are the similarities (String Distance) for Amanda's hand-selected examples? (results):

```
prefix dcterms: <http://purl.org/dc/terms/>
prefix cpdl:   <http://www3.cpdl.org/wiki/index.php/>
prefix imslp:   <http://imslp.org/wiki/>
prefix xt:     <http://purl.org/twc/vocab/cross-topix#>

SELECT ?page_1 ?page_2 ?sim

WHERE {
   GRAPH <http://leo.tw.rpi.edu/source/orange-joiner/dataset/title-similarities/version/2011-
Apr-20> {
       ?comparison xt:comparable_1 ?page_1;
              xt:comparable_2 ?page_2;
              xt:similarity   ?sim;
   }
} ORDER BY DESC(?sim)
```

What are the similarities and titles for Amanda's hand-selected examples? (results):

```
prefix dcterms: <http://purl.org/dc/terms/>
prefix cpdl:   <http://www3.cpdl.org/wiki/index.php/>
prefix imslp:   <http://imslp.org/wiki/>
prefix xt:     <http://purl.org/twc/vocab/cross-topix#>
```

```
SELECT ?page_1 ?page_2 ?sim ?title_1 ?title_2

WHERE {
  GRAPH <http://leo.tw.rpi.edu/source/orange-joiner/dataset/title-similarities/version/2011-
Apr-20> {
        ?comparison xt:comparable_1 ?page_1;
                xt:comparable_2 ?page_2;
                xt:similarity   ?sim .
  }
  GRAPH <http://leo.tw.rpi.edu/source/orange-amanda/dataset/ground-truth/version/2011-Apr-19> {
        ?page_1 dcterms:title ?title_1 .
  }
  GRAPH <http://leo.tw.rpi.edu/source/orange-amanda/dataset/ground-truth/version/2011-Apr-19> {
        ?page_2 dcterms:title ?title_2 .
  }
} ORDER BY DESC(?sim)
```

???

```
where {}
```

???

```
where {}
```

???

```
where {}
```

???

```
where {}
```

TODO: Implement Social Machine

() Load string matching algorithm results into ARC2 SPARQL endpoint
() Trial RDF with proposed triples testing with UPDATE and QUERY
() AJAX client to query page similarity results (JQuery)
() display proposed suggestions for user feedback (JQuery)
() accept and report back human response (Y/N) SPAR/UL

- Name of person
- Feedback: approve or disapprove suggestion

() learn “enough” SPARQL

- http://logd.tw.rpi.edu/technology/SPARQL lists some learning resources.

Source	Dataset	Version	Dataset URI
orange	crowd-verifications	2011-Apr-XX	2011-Apr-20

This is an example RDF that the Social Machine client will assert back to the SPARQL endpoint after the user provides a “yes” or “no” vote. TODO: Yu

```
@prefix xt:    <http://purl.org/twc/vocab/cross-topix#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

:aVote
 xt:comparison
 xt:user [ foaf:name “Text user entered” ];
 xt:vote “true”^^xsd:boolean;
.
```

SPARQL Query: To update the database to indicate whether the user thought it was a match or not. TODO - Yu.

```
# query to put vote onto endpoint
```

SPARQL Query: To get the current proposed match with the highest score, TODO

```
@prefix
```

TODO ????

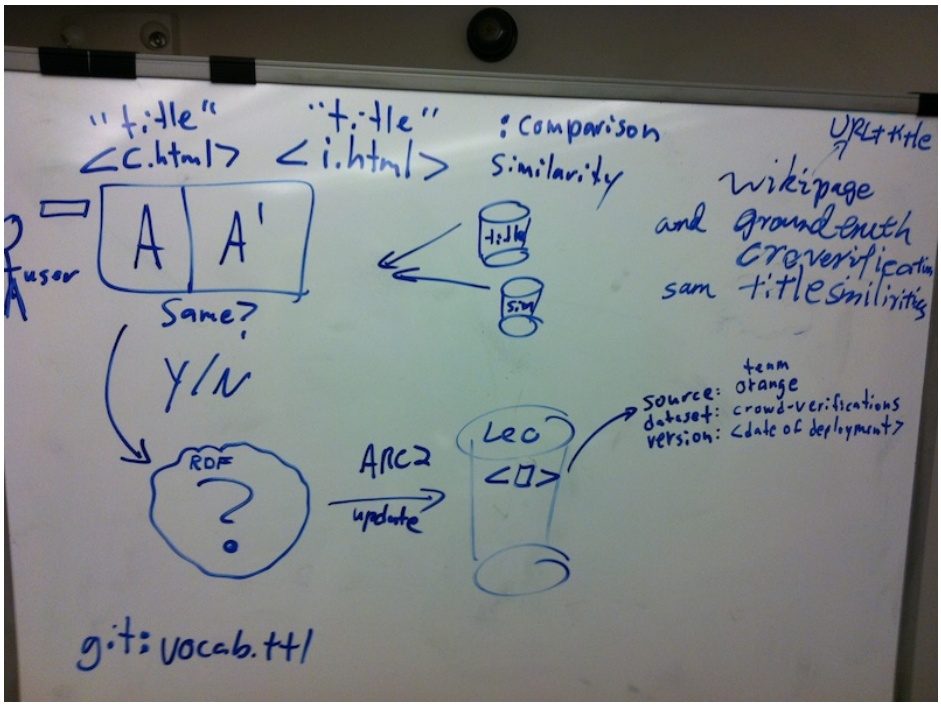


Figure: Enumerating the data attributes to consider when modeling the 1) user response, 2) submitting the response, and 3) choosing the named graph design. This discussion motivates the RDF modeling for the vote (based on the finished RDF modeling of the page titles and similarities), the cross-topix vocabulary (which was extended)

TODO: See Also box

() Accept URL of U_wiki or P_wiki
() Query “Y/N Aggregation”
() Return suggestions to alternative wiki

TODO: Write Use Cases
() Use Case for user of “social machine”
() Use Case for user of “See Also” box

TODO: Write Future Work section
() Discuss possibility of using a controlled vocabulary
() Possibility of “explicit” connection

TODO: Modeling
(X)conceptual and logical modeling (HOW is this DONE? Where is it?) add bullets.
()information architecture

TODO: Prototype Development
()Mockup
()Architectural Model

TODO:Use Case Activity Diagrams
()For Use Case 1
()For Use Case 2

TODO: Presentation Development

Test Cases

Our domain experts hand-selected matches between the two wikis.

Composers	on CPDL	on IMSLP
	Gustav_Holst	Category:Holst,_Gustav

Composers

Gustav Holst
http://www3.cpdl.org/wiki/index.php/Gustav_Holst
http://imslp.org/wiki/Category:Holst,_Gustav

Franz Liszt
http://www3.cpdl.org/wiki/index.php/Franz_Liszt
http://imslp.org/wiki/Category:Liszt,_Franz

Camille Saint-Saens
http://www3.cpdl.org/wiki/index.php/Camille_Saint-Sa%C3%ABns
http://imslp.org/wiki/Category:Saint-Sa%C3%ABns,_Camille

Pieces

Ave verum corpus, K. 618 by Wolfgang Amadeus Mozart
http://www3.cpdl.org/wiki/index.php/
Ave_verum_corpus,_KV_618_(Wolfgang_Amadeus_Mozart)
http://imslp.org/wiki/Ave_verum_corpus,_K.618_(Mozart,_Wolfgang_Amadeus)

Ich hatte viel Bekummernis, BWV 21 by Johann Sebastian Bach
http://www3.cpdl.org/wiki/index.php/Cantata_BWV_21_-
_Ich_hatte_viel_Bek%C3%BCmmerniss_(Johann_Sebastian_Bach)
http://imslp.org/wiki/
Ich_hatte_viel_Bek%C3%BCmmernis,_BWV_21_(Bach,_Johann_Sebastian)

Albendlied by Josef Rheinberger
http://www3.cpdl.org/wiki/index.php/Abendlied,_Op._69,_No._3_(Josef_Rheinberger)
http://imslp.org/wiki/Abendlied_(Rheinberger,_Josef_Gabriel)

Stabat Mater by Antonin Dvorak
http://www3.cpdl.org/wiki/index.php/Stabat_Mater_(Anton%C3%ADn_Dvo%C5%99%C3%A1k)
http://imslp.org/wiki/Stabat_Mater,_Op.58_(Dvo%C5%99%C3%A1k,_Anton%C3%ADn)

Symphony No. 2 by Gustav Mahler
http://www3.cpdl.org/wiki/index.php/Symphony_No._2_in_C_Minor_(%27Resurrection%27)_-
_choral_score_(Gustav_Mahler)
http://imslp.org/wiki/Symphony_No.2_(Mahler,_Gustav)

Requiem in D Minor by Mozart
http://www3.cpdl.org/wiki/index.php/Requiem,_KV_626_(Wolfgang_Amadeus_Mozart)
http://imslp.org/wiki/Requiem_in_D_minor,_K.626_(Mozart,_Wolfgang_Amadeus)

Grande messe des morts by Hector Berlioz

http://www3.cpdl.org/wiki/index.php/Grande_messe_des_morts,_H_75_(Hector_Berlioz)
http://imslp.org/wiki/Grande_messe_des_morts_(Requiem),_H_75_(Berlioz,_Hector)

Use Cases

"See Also" Box

What is it? Why did we make our choice?

There are two main music score wikis that are available to the public: the Petrucci Music Library and the Choral Public Domain Library. Both of these libraries have thousands of pages for individual music scores and composers. In many cases, the same pieces of music and the same composers can be found on both sites, but different information may be found on either site. We thought it would be helpful if we could provide a way for users to know that there exists this other site that contains the same information, but may be in a different format or a different edition of the publication. We decided that the best way to convey this information would be as a "See Also" box that would give suggestions to point the user towards pages or categories in the other wiki. This may expose the reader to new music or provide another source of the sheet music that may be in a more convenient format.

Use Case Name

Webpage Congruency Identifier Box

Goal

The users of the Petrucci Music Library and/or the Choral Public Domain Library will have a way to connect the information on both sites. This will not make use of a third party site, but will embed onto the current sites to provide a direct link between the two services.

Summary

The user would access the Petrucci Music Library and/or the Choral Public Domain Library to find music scores based on the search categories of composer, title, date of publication among others. Lacking viable information on one site, the user would want to find another site to satisfy their needs. For instance, the Petrucci Music Library has hand written scores which are difficult to read. Therefore, moving to the Choral Public Domain Library one might find one that is typeset. The cross-topix infrastructure is embedded on both sites to allow users to access the other site's similar listings.

Actors

Primary: Teachers
Primary: Musicians

These actors initiate the presentation of the webpage congruency identifier box after searching or browsing a page on either Wiki site.

Preconditions

- There is at least one page on the wiki that the user can view
- The sites are up and available
- The plug-in is functioning
- The user has Internet access

Triggers

- The user views a page on the wiki for a specific music score
- The user views a page on the wiki for a specific composer

Basic Flow

1. The user views a page on the first wiki.
2. A box is displayed on the page that contains links to the same information on the second wiki.
3. The user selects one of the links
4. The user is redirected to the selected page on the second wiki

Alternate Flow

1. If there is not a page on the second wiki that can be linked, a box will still be displayed, but it will not contain links to pages on the second wiki.
2. If the user does not select one of the links (User does not complete step 3), the user will remain on the first wiki and will not be redirected to the second wiki.

Post Conditions

The user is viewing either a composer page or a music score page on the second wiki.

Resources

Data:

Data	Type	Characteristics	Description	Owner	Source System
(dataset name)	Remote, In situ, Etc.	e.g. – no cloud cover	Short description of the dataset, possibly including rationale of the usage characteristics	USGS, ESA, etc.	Name of the system which supports discovery and access
Petrucci Music Library	Remote		Wiki music information site	Project Petrucci LLC	MediaWiki
Choral Public Domain Library	Remote		Wiki music information site		MediaWiki

Social Machine

What is it? Why did we make our choice?

Use Case Name

Social Machine: Crowd source information accuracy relational application

Goal

The users of the Petrucci Music Library and/or the Choral Public Domain Library will have a way to communally develop associations between the sites based on composer and scores.

Summary

The user would access the Petrucci Music Library and/or the Choral Public Domain to help create associations between the sites based on composer and score. The user would go to the “Social Machine” section. The user would then be prompted to decide whether the two links were associated through a yes or no response. This activity is designed to allow users the ability to cross reference the two sites to find the information on the sites. This allows the user input to add another layer of sophistication to creating associations. Prior to this application, string matching was the only way to cross reference the material. Through crowd sourcing it enables

better accuracy of associations.

Actors

Primary: Teachers
Primary: Musicians
Primary: Anyone involved in the community

These actors repeat answering association questions until they no longer choose to do so.

Preconditions

- Has to be pages on the Wikis to compare
- The sites are up and available
- “Social Machine” is functional
- Internet access

Triggers

- The user initiates the “Social Machine” application

Basic Flow

1. The initiates the “Social Machine”
2. The user is prompted to decide whether two web pages from separate wikis are associated with a yes or no response. The decision is based on attributes pulled from both the wikis and links to the pages for further information gathering.
3. After the choice is made, another combination is created.
4. The process is iterated until the user ceases to partake in the application.

Alternate Flow

1. There are none

Post Conditions

The user is viewing either a composer page or a music score page on the second wiki.

Resources

Data:

Data	Type	Characteristics	Description	Owner	Source System
(dataset name)	Remote, In situ, Etc.	e.g. – no cloud cover	Short description of the dataset, possibly including rationale of the usage characteristics	USGS, ESA, etc.	Name of the system which supports discovery and access
Petrucci Music Library	Remote		Wiki music information site	Project Petrucci LLC	MediaWiki
Choral Public Domain Library	Remote		Wiki music information site		MediaWiki
Association Data Store	In situ	Unmanageable	List of all possible combinations of sites between the two wikis	Orange Team	RPI Server

Installing ARC2 Triple Store and SPARQL Endpoint

Database:
TDB
ARC2(SPARQL)
How system is established:

There are several components used within this system: MySQL, Apache, PHP and ARC2.

Here is how each component are configured in steps:

MySQL:
1. Download the most comfortable version for your computer (Mine is OSX 10.6, x86, 64bit version) from MySQL http://dev.mysql.com/downloads/mysql/

2. Install according to instructions here: http://dev.mysql.com/doc/refman/5.5/en/macosx-installation-pkg.html

3. Run MySQL according to instructions here:

If you have installed the Startup Item, use this command:

```
shell> sudo /Library/StartupItems/MySQLCOM/MySQLCOM start
(ENTER YOUR PASSWORD, IF NECESSARY)
(PRESS CONTROL-D OR ENTER "EXIT" TO EXIT THE SHELL)
```

If you don't use the Startup Item, enter the following command sequence:

```
shell> cd /usr/local/mysql
shell> sudo ./bin/mysqld_safe
(ENTER YOUR PASSWORD, IF NECESSARY)
(PRESS CONTROL-Z)
shell> bg
(PRESS CONTROL-D OR ENTER "EXIT" TO EXIT THE SHELL)
```

You should be able to connect to the MySQL server, for example, by
running `/usr/local/mysql/bin/mysql'.

4. To make it more convenient in booting MySQL, it is good idea to set up alias for running command.

```
alias mysql /usr/local/mysql/bin/mysql
alias mysqladmin /usr/local/mysql/bin/mysqladmin
```

Even better, add `/usr/local/mysql/bin' to your `PATH' environment variable.

Apache and PHP:
Here is a very good tutorial, where I follow to finish installation
http://www.procata.com/blog/archives/2007/10/28/working-with-php-5-in-mac-os-x-105/

ARC2:
Here needs some command to get ARC2 working.
Assume we have successfully got MySQL, Apache and PHP working.

1. Download ARC2 from: https://github.com/semsol/arc2

2. Copy all the extracted files under the same directory where you configure the Web site directory at /etc/apache2/users. E.g. For mine machine, it is within

```
/Users/yanningchen/Sites/phpstarter
```

as shown below:

```
  GNU nano 2.0.6            File: yanningchen.conf

<Directory "/Users/yanningchen/Sites/*/">
    Options Indexes MultiViews FollowSymLinks
    AllowOverride All
    Order allow,deny
    Allow from all
</Directory>

NameVirtualHost *:80

<virtualhost *:80>
        DocumentRoot /Users/yanningchen/Sites/phpstarter
        ServerName mysites
</virtualhost>
```

So the folder where all ARC2 files should go to is:

`/Users/yanningchen/Sites/phpstarter/arc`

where “arc” is my folder that contains all extracted files of ARC2.
That’s where we get PHP server knows the fact that we have ARC2 in machine.

3. Next step is to create a database that exclusively for ARC2. We need a configuration file to include all relevant components:

config.php:

```
<?php

include_once(dirname(__FILE__).'/arc/ARC2.php'); // path to the file
ARC2.php

// SQL database configuration for storing the postings:
$arc_config = array(
  /* MySQL database settings */
  'db_host' => 'localhost',
  'db_user' => 'momo',
  'db_pwd' => '871120',
  'db_name' => 'arc2test',

  /* ARC2 store settings */
  'store_name' => 'sandbox',

  /* SPARQL endpoint settings */
  'endpoint_features' => array(
      'select', 'construct', 'ask', 'describe', // allow read
      'load', 'insert', 'delete',               // allow update
```

```
      'dump'                                          // allow backup
  ),
  'endpoint_timeout' => 60, /* not implemented in ARC2 preview */
  'endpoint_read_key' => '', /* optional */
  'endpoint_write_key' => '', /* optional */
  'endpoint_max_limit' => 250, /* optional */
);

client.php, which is responsible for database operations
#!/usr/bin/env php
<?php

include_once(dirname(__FILE__).'/config.php');

/* store instantiation */

$store = ARC2::getStore($arc_config);

if (!$store->isSetUp()) {
  $store->setUp(); /* create MySQL tables */
}

/* query handling */

$query = $argv[1];

$result = $store->query($query);

/* error handling */

if ($errors = $store->getErrors()) {
      error_log("arc2sparql error:\n" . join("\n", $errors));
      exit(10);
}

/* result handling */

if ($result["query_type"] == "construct" ||
      $result["query_type"] == "describe"
) {
      $ser = ARC2::getTurtleSerializer();
      print $ser->getSerializedIndex($result["result"]);
      print "\n";
} else if ($result["query_type"] == "select") {
      $vars = $result['result']['variables'];
      $rows = $result['result']['rows'];
      foreach ($vars as $var) {
      print $var . " ";
```

```
      }
      print "\n";
      foreach ($rows as $row) {
      foreach ($vars as $var) {
            print $row[$var] . " ";
      }
      print "\n";
      }
} else if ($result["query_type"] == "load") {
      print "Loaded " . $result["result"]["t_count"] . " triples.\n";
} else if ($result["query_type"] == "insert") {
      print "Inserted " . $result["result"]["t_count"] . " triples.\n";
} else if ($result["query_type"] == "delete") {
      print "Deleted " . $result["result"]["t_count"] . " triples.\n";
} else if ($result["query_type"] == "ask") {
      if ($result["result"]) {
      print "yes\n";
      exit(0);
      } else {
      print "no\n";
      exit(1);
      }
} else if ($result["query_type"] == "dump") {
      // The query already printed the dump, nothing to do here
} else {
      // Something unexpected
      var_dump($result);
}

exit(0);

?>
```

And endpoint.php, which handles SPARQL endpoints

```
<?php

include_once(dirname(__FILE__).'/config.php');

/* instantiation */
$ep = ARC2::getStoreEndpoint($arc_config);

if (!$ep->isSetUp()) {
  $ep->setUp(); /* create MySQL tables */
}

/* request handling */
$ep->go();
```

```
?>
```

Put this all these 3 files under the same directory of index.php

4. Now we can create the database under command line

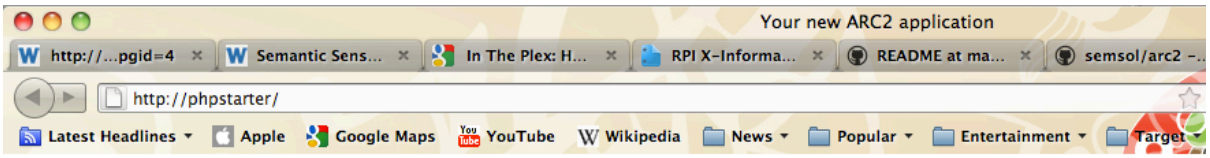
```
$> mysql -h localhost -u config_db_user -p -e "create database
config_db_name;"
```

Where config_db_user is the user name configured in the config.php file, config_db_name is the name of the database in the config.php file

5. And then, you can run cli.php on commond line and see results from the localhost. For example:

```
- chmod +x cli.php
- ./cli.php "LOAD <http://chatlogs.planetrdf.com/swig/2009-07-26>"
```

And the we see



Welcome to your new ARC2 application

- This installer and application template for the ARC2 RDF store is `arc2-starter-pack` developed at GitHub.
- Home at DERI Linked Data Research Centre

Getting started

1. First, you need to **finish the installation**.
2. After that, you can start to use ARC2 with SPARQL queries and SPARQL+ commands:
   - Access on the Web via **your SPARQL endpoint**.
   - Access on the command line via **`cli.php`**, e.g.:
     - `cd /Users/yanningchen/Sites/phpstarter/`
     - `chmod +x cli.php`
     - `./cli.php "LOAD <http://chatlogs.planetrdf.com/swig/2009-07-26>"`
     - `./cli.php "LOAD <file:///home/user/local_file.rdf>"`
     - `./cli.php "LOAD <file://$PWD/file_in_current_dir.ttl>"`
     - `./cli.php "SELECT DISTINCT ?property WHERE { ?subject ?property ?object . }"`
     - `./cli.php "DELETE FROM <http://chatlogs.planetrdf.com/swig/2009-07-26>"`

Developing this into your own PHP application

You can **edit this `index.php`** to become your own application. Here's a start:

Running PHP code...

Properties currently in use in the triple store
http://www.w3.org/1999/02/22-rdf-syntax-ns#type
http://xmlns.com/foaf/0.1/chatEventList
http://purl.org/dc/elements/1.1/date
http://purl.org/dc/elements/1.1/description

And we done.

Database:
Username:orange
Password:orange
Databasename:orangedatabase

Future Work

Future List
1. User scoring evaluation- protection for site

2. What next algorithm – decides what potential match the user sees in order to continually evaluate the matches and the user selections

Better similarity measures. Although we were able to get measure better than direct string similarity, there is a wealth of heuristics that could be applied or developed in future iterations. For example, analyzing the narrative in the document itself or use the connectivity among the pages could be used to create different and better measures for similarity.

Multiple measure types leads to handling multiple similarity measures. Would require a modeling change to associate the Comparison to a pairing of the quantity with the technique that produced it. We currently have the similarity attribute citing the quantity directly with an implicit understanding that it uses our default similarity heuristic.

Similarity Measures